Weighing in on Issues with “Cloud Scale”

Michael Coppola

Ekoparty 2013
Who am I?

- #
- Student at Northeastern University, USA
- CTF every now and then
- http://poppopret.org/
In this presentation

- A bit of this
- A bit of that
- Successes, failures
- Tool development
- Mad t3chn1quez
Cloud scaling

OMG CLOUD
Wireless Scale WS-30

Step up for instant weighing and BMI.
Attack surface

- WiFi / Bluetooth driver and application
- Network communications
- Application input parsing
- No network services (no open ports)
Can the Wireless Scale be kept up-to-date with new software?

Yes, Wireless Scale is a smart and updateable device. It can be updated with new software to add new features, make it compatible with new apps or devices, or fix issues that our users have reported. If you have a Wi-Fi network, software update will occur automatically at night as soon as an update is available. If you don't have a Wi-Fi network, the Withings app will advise you when an update is available and provide update instructions.

99% chance this was implemented horribly
Sniffing network traffic

- Associate to WiFi using config from Bluetooth
- DNS lookup for scalews.withings.net
- JSON-based protocol over plaintext HTTP
Sniffing network traffic

- Challenge-handshake authentication
- Send device info (MAC, fw version, battery)

POST /cgi-bin/once HTTP/1.1
action=get

HTTP/1.1 200 OK
{
  "status": 0,
  "body": {
    "once": "00d016bf-242e0bb1"
  }
}

POST /cgi-bin/session HTTP/1.1
action=new
&auth=00:24:e4:06:59:dc
&hash=25fd29132cf66a5cdf1a7efdc673be26
&mfgid=262151&currentfw=211
&batterylvl=69&duration=30&zreboot=1
MITM'ing network traffic

- We want the firmware image
  - Maybe sending a lesser fw version will initiate an update

- We don't know how to complete the handshake, so we still need the device

- DNS spoof the device, interpose ourselves in the session
DNS spoofing the handshake

1. Device initiates connection
2. Server responds with nonce
3. Device sends calculated hash with diagnostic info

3'. Hacker modifies the fw version and sends to the server
The response

{"status":0,"body":{"sessionid":"8051-51492c4d-730e4ff3","sp":
{"users":[]},"ind":{"lg":"en_GB","imt":1,"stp":1,"f":0,"g":97918},"syp":
{"blc":"http://fw.withings.net/wbs03_211.bin","utc":1363749965},"ctp":
{"goff":-14400,"dst":0,"ngoff":0}}}
Firmware header

- No results from binwalk
- Lots of strings → likely no encryption or compression
- Multiple null padded sections → likely multiple objects packaged together

```
$ hexdump -C wbs03 211.bin | head
00000000 7c 47 0a 00 01 00 00 00      d3 00 00 00 28 00 00 00                       |G...................(...
00000010 f0 61 06 00 65 f6 3b 0c      18 62 06 00 a4 11 03 00                       |.a..e.;..b......|
00000020 bc 73 09 00 ba d3 00 00      00 00 01 20 e9 91 02 00                       |.s..................|
00000030 cd 91 02 00 19 96 02 00      cd 91 02 00 cd 91 02 00                       |..................|
00000040 cd 91 02 00 cd 91 02 00      cd 91 02 00 cd 91 02 00                       |..................|
00000050 cd 91 02 00 05 58 04 00      cd 91 02 00 cd 91 02 00                       |..................|
00000060 d5 58 04 00 11 59 04 00      45 8b 02 00 6d 8b 02 00                       |.X...Y..E...m...|
00000070 ed 3a 01 00 cd 91 02 00      cd 91 02 00 cd 91 02 00                       |..................|
00000080 cd 91 02 00 cd 91 02 00      cd 91 02 00 cd 91 02 00                       |..................|
```
Identifying the MCU

- MK20DN512ZVLL10
- Freescale Kinetis K20 family
- ARM Cortex-M4 (ARMv7)
- Memory-mapped peripheral registers
Locating code blocks

- Find a dense area of bytes and start disassembling
  
  - Common bytes:
    - ARM: 0xe*
    - Padding: 0xbf00 (nop)
  
- Byte search the addresses of strings and disassemble backwards
Things aren't lining up...

```
loc_37A7A
LDR.W R4, [R8,#0x268]
STR R4, [R7,#0x4C+var_28]
CBNZ R4, loc_37A9C

LDR.W R3, [R8,#0x24C]
CBNZ R3, loc_37A98

LDR R3, =aT_inputRead bu ; "t_input: Read_Buffer_Size command failed...
LDR R2, =unk_68A46
LDR R4, [R7,#0x4C+var_34]
CMP R4, #2
ITE NE MOVME R2, R3
STR R2, [R7,#0x4C+var_26]
B loc_37A9C

loc_37A98
LDR R4, =unk_66E5
STR R4, [R7,#0x4C+var_28]

loc_37A9C
MOV R0, R8
LDR R1, =unk_66F31
BL sub_37462
CBZ R7, R0, loc_37A9E
```

```assembly
ROM: 00050133 5B
ROM: 00050134 46
ROM: 00050135 43
ROM: 00050136 49
ROM: 00050137 5D
ROM: 00050138 20
ROM: 00050139 68
ROM: 0005013A 63
ROM: 0005013B 69
ROM: 0005013C 5F
ROM: 0005013D 65
ROM: 0005013E 76
ROM: 0005013F 65
ROM: 00050140 6E
ROM: 00050141 74 5F 69 6E+aT_inputRead bu DCB "t_input: Read_Buffer_Size command failed, 0x%0x %s",0xA,0
ROM: 00050141 70 75 74 3A+
ROM: 00050141 20 52 65 61
```
- Every dword in file is treated as a pointer

- Does base + dword point to the beginning of a string?

- Repeat for all possible base addresses

- Highest score is likely the correct base address
One room available in Huaihai xi lu - Panyu Lu

Hi everybody!
I'm leaving shanghai and my room will be available on the 21th of October.
It's a nice room in a 4 bedroom shared appartement located in Huai Hai xi road and panyu Road.
You'll be sharing this appartement with one HongKongnese girl, one latvian guy and one french guy.
The flat has a spacious living room, a balcony with a nice view on a small park, a full kitchen, and all needed services: Washing machine, TV, DVD player, wireless intemet, Air conditionner for every room and for the living room. An api is passing by twice a week to clean the flat.
The appartement is very convenient in term of location:
- 5 min from Hongqiao Lu subway station, line 3/4/10
- 15 min walk from Xujiabui subway station, line 1
- 1 min walk from bus 911, 926 which lead you directly to people square
- 15 min from Grand Gateway, Xujiabui
Send me a message to my personal mail fdusanter@gmail.com if you are interested!
Firmware subsystems

- LibCURL – HTTP library
- lwIP – TCP/IP stack
- DbLib – Key=value store
- WsLib – Interact with Withings web services
- CnLib – Network connection management
- UsLib – User management
- LibPairing – Bluetooth pairing
### DbLib_GetElement() Table

<table>
<thead>
<tr>
<th>Index</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>mac_address</td>
<td>0x12</td>
</tr>
<tr>
<td>2</td>
<td>hostname</td>
<td>0x40</td>
</tr>
<tr>
<td>3</td>
<td>secret</td>
<td>0x11</td>
</tr>
<tr>
<td>4</td>
<td>time_constant</td>
<td>0xc</td>
</tr>
<tr>
<td>5</td>
<td>users</td>
<td>0x280</td>
</tr>
<tr>
<td>6</td>
<td>uid</td>
<td>0x26</td>
</tr>
<tr>
<td>7</td>
<td>pref_lang</td>
<td>0x6</td>
</tr>
<tr>
<td>8</td>
<td>scale_constant</td>
<td>0x10</td>
</tr>
<tr>
<td>9</td>
<td>ssid</td>
<td>0x6c</td>
</tr>
<tr>
<td>10</td>
<td>IP config</td>
<td>0x16</td>
</tr>
<tr>
<td>11</td>
<td>mfg_id</td>
<td>0x4</td>
</tr>
<tr>
<td>12</td>
<td>calibration</td>
<td>0x14e</td>
</tr>
<tr>
<td>14</td>
<td>last connection</td>
<td>0x20</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>0x2c</td>
</tr>
<tr>
<td>17</td>
<td>factory_mode</td>
<td>0x4</td>
</tr>
<tr>
<td>19</td>
<td>debug traces</td>
<td>0x0</td>
</tr>
<tr>
<td>20</td>
<td>factory weight verif</td>
<td>0x4c</td>
</tr>
<tr>
<td>22</td>
<td>wifi_country</td>
<td>0x4</td>
</tr>
<tr>
<td>24</td>
<td>wpa_key</td>
<td>0x50</td>
</tr>
<tr>
<td>25</td>
<td>battery level</td>
<td>0x8</td>
</tr>
<tr>
<td>27</td>
<td>wifi_delay</td>
<td>0x4</td>
</tr>
<tr>
<td>28</td>
<td>Bluetooth config</td>
<td>0x16</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>0x4</td>
</tr>
<tr>
<td>33</td>
<td>calibration parameters</td>
<td>0x4e</td>
</tr>
<tr>
<td>34</td>
<td></td>
<td>0x10</td>
</tr>
<tr>
<td>37</td>
<td></td>
<td>0x4</td>
</tr>
<tr>
<td>38</td>
<td></td>
<td>0x82</td>
</tr>
</tbody>
</table>
Device association request

```
printf("[WSLIB] In StartSession, once <\%s>\n", &once);
DbLib_GetElement(1, mac_addr, 0x12);
DbLib_GetElement(3, secret, 0x11);
sprintf(resbuf, "%s:%s:%s", mac_addr, secret, &once);
printf("resbuf = %s\n", resbuf);
hash = get_hash(resbuf);
mfgid = get_mfg_id();
currentfw = get_firmware_version();
batterylvl = get_battery_level();
zreboot = get_zreboot();
res = snprintf(
    POST_fields,
    220,
    "action=new&auth=%s&hash=%s&mfgid=%d&currentfw=%d&batterylvl=%d&duration=30&zreboot=%d",
    mac_addr,
    hash,
    mfgid,
    currentfw,
    batterylvl,
    zreboot);
if ( res > 219 )
    bof_detected = ((unsigned int)(res + 1) <= 0) | 1;
else
    bof_detected = (unsigned int)(res + 1) <= 0;
if ( bof_detected )
{
    v18 = "$[WSLIB] StartSession (new) buffer overflow\n"
LABEL_29:
    printf(v18);
    goto LABEL_30;
}
if ( do_HTTP_POST_request(hostname, "session", POST_fields, resbuf, 0x767) )
{
    err_msg 0 = "$[WSLIB] StartSession (new) HTTP error\n"
    goto LABEL_26;
}
sub_168EC();
v19 = new_JSON_parser(&config);
if ( !v19 )
{
    v18 = "$[WSLIB] StartSession JSON mem error\n"
    goto LABEL_29;
}
```
Cracking the firmware header

```
[mncoppola@dysthymia firmware]$ hexdump -C wbs03_211.bin | head -n3
00000000  7c 47 0a 00 01 00 00 00  d3 00 00 00 28 00 00 00  ||G...........(...
00000010  f0 61 06 00 65 f6 3b 0c  18 62 06 00 a4 11 03 00  |.a..e;...b.....
00000020  bc 73 09 00 ba d3 00 00  00 00 01 20 e9 91 02 00  |.s.............
```

```
[mncoppola@dysthymia firmware]$ hexdump -C wbs03_211.bin | tail -n3
000a4760  d1 1c 60 01 e0 13 68 0b  60 14 60 10 bd 00 00 4e  |...`..h.`..`....N|
000a4770  fc 04 ff ff ff ff ff ff 65 68 f3 4f  |.........eh.o|
```

```c
if ( open_spi_flash(&SPI0_regs_base) )
{
    printf("Fail to open flash\n");
}
else
{
    memset(&fw, 0, 0x128U);
    read_from_SPI(&SPI0_regs_base, (char *)&fw, bank_addr, 0x28);
    sub_235D00(&SPI0_regs_base);
    snprintf(str, 0x20, "blk%d.tbl_", op);
    printf("%s\n\n", str, fw.total_size);
    printf("%s\n", str, fw0.gold);
    printf("%s\n", str, fw0.version);
    printf("%s\n", str, fw0.kinetics_address);
    printf("%s\n", str, fw0.kinetics_size);
    printf("%s\n", str, fw0.bluetooth_crc);
    printf("%s\n", str, fw0.bluetooth_size);
    printf("%s\n", str, fw0.bluetooth_address);
    ret = check_firmware_CRC32(crc_op, &crc, &tmp);
    printf("%s\n", str, fw0.bluetooth_crc);
    *(DWORD *)&valid = "no";
    if (!ret )
    {
        *(DWORD *)&valid = "yes";
        printf("%s\n", op, *(DWORD *)&valid);
    }
    LOL:
    JUMPOUT(_CS_, fw0.kinetics_crc);
} 
```
Reversing the CRC validation

```c
unsigned int __fastcall init_CRC32(unsigned int poly, unsigned int seed)
{

    SIM_SCGC6 |= 0x400000u;
    CRC_CTRL |= 0x10000000u; // TCRC = 1, 32-bit CRC mode
    CRC_CTRL |= 0x40000000u; // FXOR = 1, complement data
    CRC_CTRL |= 0x20000000u; // TOTR = 10, bits and bytes are transposed for read
    CRC_CTRL |= 0x80000000u; // TOT = 10, bits and bytes are transposed for write
    CRC_GPOLY = poly;
    CRC_CTRL |= 0x20000000u; // WAS = 1, writes to CRC_CRC (data register) are seed values
    CRC_CRC = seed;
    CRC_CTRL &= 0xFDFFFFFF;
    CRC_CRC = seed;

    return poly;
}
```

```c
    read_from_SPI(&SPI0_regs_base, firmware, bank_addr, 160);
    offset = *(DWORD *)firmware - 4; // first dword is firmware size
    if ( (unsigned int)(*(DWORD *)firmware - 4) <= 0xDFFF0) // max firmware size 917,500 bytes
    {
        read_from_SPI(&SPI0_regs_base, (char *)&firmware_CRC, bank_addr + offset, 4); // get firmware checksum
        init_CRC32(CRC32_POLY, 0xFFFFFFFF);
        for ( i = 0; ; i = next_i )
        {
            next_i = i + 512;
            if ( i + 512 > (unsigned int)offset )
                break;
            read_from_SPI(&SPI0_regs_base, fw_data, bank_addr + i, 512);
            sub_311F0();
            write_to_CRC(fw_data, 128);
        }
        leftover = offset - i;
        if ( leftover > 0 )
        {
            read_from_SPI(&SPI0_regs_base, fw_data, bank_addr + i, (unsigned __int16)leftover);
            write_to_CRC(fw_data, leftover >> 2);
        }
    }
    calculated_CRC = CRC_CRC;
```
Crafting arbitrary images

mncoppola@dysthymia:~/Desktop/withings/firmware$ ./image_chksum modified_211.bin out.bin
Image is 673660 bytes
Found WS-30 firmware image:
  Total size: 673660
  Gold: 0x00000001
  Version: 211
  Kinetis address: 0x28
  Kinetis size: 418288
  Kinetis CRC: 0x0c3bf665
  WiFi address: 0x66218
  WiFi size: 201124
  Bluetooth address: 0x973bc
  Bluetooth size: 54202
  Image CRC: 0x4ff36865

Calculated image CRC: 0x709f7f98
Calculated Kinetis CRC: 0x88de9b69

Patched Kinetis CRC, recalculating image CRC...
Calculated image CRC: 0x16475d28
Patched image CRC

Wrote 673660 bytes to out.bin
Let's fuzz this thing

- We want some delicious 0dayz
- Send it invalid JSON / garbage values
- We need introspection
- Debug console?
Debug console!
8.1 K20 Signal Multiplexing and Pin Assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT functionality is available on each pin.
Spot the UART!
Not a great signal...
Introspection-less

• Code execution on the device would solve all the problems

• Hook the ARM hard fault handler to send crash dumps over the wire

• Re-purpose WsLib to easily send HTTP requests with register context info
struct processor_status {
    unsigned int r0;
    unsigned int r1;
    unsigned int r2;
    unsigned int r3;
    unsigned int r12;
    unsigned int lr;
    unsigned int pc;
    unsigned int psr;
};

#define sprintf ((int (*)(char *, char *, ...))0x47915)
#define do_HTTP_POST_request ((int (*)(char *, char *, char *, char *, int))0x16425)

void hard_fault_handler ( void )
{
    /* All function pointers +1 for thumb */
    char buf[100];
    struct processor_status *regs;

    sprintf(buf, 0x5c22c, regs->r0, regs->r1, regs->r2, regs->r3, regs->r12, regs->lr, regs->pc, regs->psr);

    do_HTTP_POST_request("hacker", "fault", buf, 0, 0);
}
Pushing updates

• We'll probably brick it at some point

• Need a recovery plan

• No bootloader we can break into

• Dump (and reprogram) the flash memory!
Identify the flash chip
Desolder the flash chip

- Heat gun + tweezers
- Soldering iron blade tip
- Solder wick
- Rework station
This means you did it wrong
This means you fixed it wrong
Fuck it, new scale
Replace chip with pins
Yup, it still works
Score!

<table>
<thead>
<tr>
<th>mac_addr</th>
<th>secret</th>
<th>nonce</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:24:e4:06:59:dc</td>
<td>649a16bf977d3b3e</td>
<td>00d016bf-242e0bb1</td>
</tr>
</tbody>
</table>

**Challenge format:**
<mac_addr>:<secret>:<nonce>

**MAC Address:** 00:24:e4:06:59:dc
**Secret:** 649a16bf977d3b3e
**Nonce:** 00d016bf-242e0bb1

**HASH:** 25fd29132cf66a5cdf1a7efdc673be26
DEMO TIME
Lessons learned

- ARM compilers are aggressive
  - Reference middle of strings
  - Inline everything possible, even data

- Strings are your friends
  - Find base address
  - Find code blocks
  - Determine symbol names, branch purposes, debugging info

- Lots of help from hardware data sheets and reference manuals

- Embedded system security sucks
Thanks

- Albert Cahalan
- Rob Jerrell
- Jordan Wiens
- Andrew Watts
- Paul Furtado
Hi Michael,

I just wanted to touch based on behalf of the Withings tech team as all were impressed with your reverse engineering work on our scale. You did a great job and as a reward we would be pleased to send you a brand new unit if you want one - just let me know!

Best,

Alexis

--
Alexis Arquillière
Product Manager
Withings | Turn on your well-being
alexis.arquilliere@withings.com | +33 613 831 080

www.withings.com  b: blog.withings.com
t: www.twitter.com/withings  fb: www.facebook.com/withings
Greetz

- #busticati
- Marauders
- bliss, thing2
Questions?

@mncoppola
poppopret.org